Random Finite Set for Multi-object Dynamical System

Ba-Ngu Vo
ba-ngu.vo-au.com/

Department of ECE
Curtin University
Perth Western Australia

Matlab code
ba-tuong.vo-au.com/
Classical (dynamical) system

- System state evolves in the state space
- States are hidden & only partially observed in the observation space
- Fundamental (dynamical) system problems:
  - Filtering
  - System ID
  - Control
Introduction

1940’s: Wiener filter
Pioneering work by Wiener, Kolmogorov

1950’s: Kalman filter
Work by Bode & Shannon, Zadeh & Ragazzini, Levinson, Swerling, Stratonovich, etc.

1960’s:
Publication of the Kalman filter, Kalman-Bucy filter, Schmidt’s 1st implementation – Apollo program
LMS algorithm by Widrow & Hoff

1970’s: Aerospace applications
Sorenson & Alspach, Singer, Bar-Shalom, Reid, etc.
Introduction

- **Particle Filter (1990’s--)**
  Computational tools for non-linear filtering
  Gordon, Salmond & Smith,
  Doucet et al, Ristic, et al …

- **Multi-Object Filter (1990’s--)**
  Unified framework for multi-sensor multi-object
  information fusion, by Mahler …
Introduction

Dynamical systems with multiple states

- Particles
- Molecules
- Cells

- People
- Orbital Debris
- Astronomy
Introduction

Dynamical systems with multiple states

- **Applications**: target tracking, computer vision, robotics, biomed, …
- **Fundamental difference from classical dynamical system**:
  - Random number of states (and possibly measurements)
  - Need new theories + solutions for filtering, system ID, and control
Random Finite Set for Multi-object Dynamical System

- Introduction
- Review of Estimation Theory
  - Random variables
  - Estimator
  - State space models
  - Bayes filter
    - Kalman filter
    - Particle filter
- Multi-object Dynamical System
  - Multi-object (system) state
  - Multi-object (system) trajectory
  - Multi-object error metric
Random Finite Set for Multi-object Dynamical System

- Bayesian Multi-object Estimation
  - Random Finite Set
  - Probability Density of an RFS
  - Multi-object Estimators
  - Multi-object State Space Model
  - Multi-object Filtering

- Generalized Labeled Multi-Bernoulli (GLMB) Filter
  - Labeled RFS
  - GLMB
  - GLMB Filter
    - GLMB Prediction Truncation
    - GLMB Update Truncation
    - Joint GLMB Prediction and Update
Random Finite Set for Multi-object Dynamical System

- Recent Research in Multi-object Systems
  - Large-scale GLMB Filter
  - GLMB Smoothing
  - RFS POMDP
  - Track-Before-Detect
  - Tracking with Merged Measurements
  - Multiple Extended Targets
  - SLAM
  - Visual Tracking
  - Filtering with Multiple Spawning Objects
Random Variables

Formal definition: Need a sample space $\Omega$ & a space $\mathbb{X}$ (e.g. $\mathbb{R}^n$)

An $\mathbb{X}$-valued **random variable (RV)** $X$ is a mapping $X : \Omega \rightarrow \mathbb{X}$

The **probability distribution** $P_X$, of $X$, is defined for each $\mathcal{T}$ by

$$P_X(\mathcal{T}) = \text{Prob}(X^{-1}(\mathcal{T}))$$
As far as we are concerned:

A Random Variable $X$ is described by a **probability distribution** $P_X$, or (in many applications) a **probability density function** (PDF) $p_X$

- **probability distribution**

  \[ P_X(\mathcal{T}) = \text{Probability that } X \text{ lies in } \mathcal{T} \]

- **PDF**

  \[ p_X(x) = \text{Likelihood that } X \text{ takes on the value } x \]

  \[ P_X(\mathcal{T}) = \int_{\mathcal{T}} p_X(x) \, dx \quad (\text{Assuming } \mathbb{X} = \mathbb{R}^n) \]
Random Variables

- Probability distribution & PDF on Euclidean space

\[ P_X(T) = \int_T p_X(x) dx \]

\[ p_X(x) = \lim_{\Delta x \to 0} \frac{P_X(\Delta x)}{Vol(\Delta x)} = \frac{P_X(dx)}{dx} \]

= Probability per unit volume

has a unit e.g. \( m^3 \)

Generalisation: Replace volume by a measure \( \mu \) e.g. another probability distribution

\[ p_X(x) = \lim_{\mu(\Delta x) \to 0} \frac{P_X(\Delta x)}{\mu(\Delta x)} = \frac{P_X(dx)}{\mu(dx)} \]

\[ \int_T P_X(dx) = \int_T p_X(x) \mu(dx) \]

- Expectation of an RV \( \mathbb{E}[f(X)] = \int f(x) p_X(x) dx \)
Random Variables

- **Joint random variable** $X,Y$
  
  $$p_{X|Y}(x \mid y) = \frac{p_{XY}(x, y)}{p_Y(y)}$$

  **Independence**: The RVs $X, Y$ are (statistically) independent if
  
  $$p_{XY}(x, y) = p_X(x)p_Y(y)$$

  **Bayes Rule**:

  $$p_{Y|X}(y \mid x) = \frac{p_{X|Y}(x \mid y)p_Y(y)}{p_X(x)}, \quad p_X(x) > 0$$

  **Total Probability**:

  $$p_X(x) = \int p_{XY}(x, y)dy$$

  $$p_X(x) = \int p_{X|Y}(x \mid y)p_Y(y)dy$$
In an estimation problem we are given the data $z$.

Data model: $p(z \mid \theta) = \text{pdf of data } z, \text{ given parameter } \theta$

\textbf{Likelihood function}

\textbf{Parameter} $\theta$ is unknown and is to be estimated.

An \textbf{estimator} is a function of the data $z$.

$$\hat{\theta} : z \mapsto \hat{\theta}(z) \in \text{parameter space}$$

\textbf{Estimation error}: measures how close the estimate is from the truth

$$\hat{\theta}(z) - \theta$$

- Non-Bayesian: treats $\theta$ as a deterministic quantity
- Bayesian: treat $\theta$ as a realisation of a random quantity
## Estimator

<table>
<thead>
<tr>
<th>Type</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum likelihood (ML)</td>
<td>( \hat{\theta}(z) = \hat{\theta}^{ML}(z) = \arg \max_{\theta} p(z</td>
</tr>
<tr>
<td>Maximum A Posteriori (MAP)</td>
<td>( \hat{\theta}(z) = \hat{\theta}^{MAP}(z) = \arg \max_{\theta} p(\theta</td>
</tr>
<tr>
<td>Minimum Mean Square Error (MMSE)</td>
<td>( \hat{\theta}^{MMSE}(z) = \arg \min_{\hat{\theta}} E\left[ |\hat{\theta} - \theta|^2</td>
</tr>
<tr>
<td>Least Square (LS)</td>
<td>( \hat{\theta}(z) = \hat{\theta}^{LS}(z) = \arg \min_{\theta} |z - h(\theta)|^2 )</td>
</tr>
</tbody>
</table>

\( z \) is a realisation of a random process \( w \) is a realisation of a random process.
Estimator

- The posterior contains all relevant information about the parameter based on:
  - the given data,
  - data model and
  - prior information on the data

- Given the posterior, what is the “optimal” estimate of the parameter?

**Bayes Risk:** Expected posterior cost/penalty of incorrect estimate

\[
R(\tilde{\theta}(z) \mid z) = \mathbb{E}[C(\tilde{\theta}(z), \theta) \mid z]
\]

- Conditional Bayes risk of \( \tilde{\theta}(z) \)
- Cost of using \( \tilde{\theta}(z) \) as an estimate of \( \theta \)

\( C \) satisfies:
\[ C(x, y) \geq 0 \]
\[ C(x, x) = 0 \]

Example: \( C(x, y) = \| x - y \|^2 \)

Overall Bayes risk of \( \tilde{\theta} \):
\[
R(\tilde{\theta}) = \mathbb{E}[R(\tilde{\theta}(z) \mid z)] = \int R(\tilde{\theta}(z) \mid z) p(z)dz
\]
Estimator

e.g. \( C(\tilde{\theta}, \theta) = \|\tilde{\theta} - \theta\|_2^2 \)

\[ C(\tilde{\theta}, \theta) = \|\tilde{\theta} - \theta\|_1 \]

\[ C_{\Delta}(\tilde{\theta}, \theta) = \begin{cases} 
1, & \|\tilde{\theta} - \theta\|_1 > \Delta \\
0, & \|\tilde{\theta} - \theta\|_1 \leq \Delta 
\end{cases} \]

\[ \frac{R(\tilde{\theta} \mid z)}{\Delta} = \frac{1}{\Delta} - p(\tilde{\theta} \mid z) \]

\[ \hat{\theta}^{Bayes} (z) = \hat{\theta}^{MMSE} (z) = \mathbb{E}[\theta \mid z] \]

\[ \int_{-\infty}^{\hat{\theta}^{Bayes} (z)} P(d\theta \mid z) = 0.5 \]

\[ \lim_{\Delta \to 0} \hat{\theta}_{\Delta}^{Bayes} (z) = \hat{\theta}^{MAP} (z) \]
Optimal Bayes estimator: the estimator

\[ \hat{\theta}^{Bayes} : z \mapsto \hat{\theta}^{Bayes}(z) = \arg \min_{\hat{\theta}} R(\hat{\theta} \mid z) \]

minimises the overall Bayes risk \( R \) (over the set of all estimators)

\[ R(\hat{\theta}) = \mathbb{E}[R(\hat{\theta}(z) \mid z)] = \int R(\tilde{\theta}(z) \mid z) p(z)dz \]

- The posterior mean \( \mathbb{E}[\theta \mid z] \) is Bayes optimal for \( C(x, y) = \| x - y \|^2 \)
State-Space Models

- We only consider discrete-time systems
- State dynamics: modelled by difference equation e.g.

\[ \text{current position} = \text{previous position} + \text{velocity} \times \text{time} \]

- Randomness in the motion (due to wind, turbulence …)
- State vector: e.g.

\[
x_k = \begin{bmatrix}
p_{x,k} \\
v_{x,k} \\
p_{y,k} \\
v_{y,k}
\end{bmatrix}
\]

\[ X \rightarrow \text{position} \]
\[ Y \rightarrow \text{velocity} \]

Random walk model: \( x_k = x_{k-1} + \text{noise} \)
(Nearly) Constant Velocity model:

\[
\begin{bmatrix}
    p_{x,k} \\
    v_{x,k} \\
    p_{y,k} \\
    v_{y,k}
\end{bmatrix} = \begin{bmatrix}
    p_{x,k-1} + \Delta_k v_{x,k-1} \\
    v_{x,k-1} \\
    p_{y,k-1} + \Delta_k v_{y,k-1} \\
    v_{y,k-1}
\end{bmatrix} + \begin{bmatrix}
    1 & \Delta_k & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & \Delta_k \\
    0 & 0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
    p_{x,k-1} \\
    v_{x,k-1} \\
    p_{y,k-1} \\
    v_{y,k-1}
\end{bmatrix} + \text{noise}
\]

Sampling interval

\[x_k = F_{k|k-1} x_{k-1} + \text{noise}\]

\[\text{noise} \sim \mathcal{N}(\cdot; 0, Q_k)\]

\[Q_k = \begin{bmatrix}
    \Delta_k^3 / 3 & \Delta_k^2 / 2 & 0 & 0 \\
    \Delta_k^2 / 2 & \Delta_k & 0 & 0 \\
    0 & 0 & \Delta_k^3 / 3 & \Delta_k^2 / 2 \\
    0 & 0 & \Delta_k^2 / 2 & \Delta_k
\end{bmatrix}\]
General dynamic models:

\[ x_k = \Phi_{k|k-1}(x_{k-1}, v_{k-1}) \]

pdf of \( x_k \) given \( x_{k-1} \)

Markov transition density

e.g. if \( v_k \) is zero-mean Gaussian with covariance \( Q_k \)

\[ f_{k|k-1}(x_k | x_{k-1}) = \mathcal{N}(x_k; \Phi_{k|k-1}(x_{k-1}), Q_k) \]
The state $x_k$ is often hidden, only partial information $z_k$ about the state is observed by the sensor, e.g.

- **Position sensor**

$$z_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x_k + \text{noise}$$

- **Bearings and range sensor**

$$z_k = \begin{bmatrix} \theta_k \\ r_k \end{bmatrix} = \begin{bmatrix} \arctan \left( \frac{p_{x,k} - s_{x,k}}{p_{y,k} - s_{y,k}} \right) \\ \sqrt{\left( p_{x,k} - s_{x,k} \right)^2 + \left( p_{y,k} - s_{y,k} \right)^2} \end{bmatrix} + \text{noise}$$
Other sensor observation
- Time difference of arrival (TDOA)
- Frequency difference of arrival (FDOA)
- TDOA/FDOA pair
- Power

General observation model

\[
z_k = \Psi_k(x_k, w_k)
\]

\[
g_k(z_k | x_k)
\]

pdf of \(z_k\) given \(x_k\)

Observation likelihood function
e.g. if $w_k$ is zero-mean Gaussian with covariance $R_k$

$$g_k(z_k | x_k) = \mathcal{N}(z_k; \Psi_k(x_k), R_k)$$
State-Space Models

- State space model = dynamic model & observation model

\[
\begin{align*}
x_k &= \Phi_{k|k-1}(x_{k-1}, v_{k-1}) \\
z_k &= \Psi_k(x_k, w_k)
\end{align*}
\]

\[
\begin{align*}
f_{k|k-1}(x_k|x_{k-1}) \\
g_k(z|x_k)
\end{align*}
\]

- Linear Gaussian state space model: linear transformations & Gaussian noise

\[
\begin{align*}
x_k &= F_{k|k-1}x_{k-1} + v_{k-1} \\
z_k &= H_kx_k + w_k
\end{align*}
\]

\[
\begin{align*}
f_{k|k-1}(x_k|x_{k-1}) &= \mathcal{N}(x_k; F_{k|k-1}x_{k-1}, Q_{k-1}) \\
g_k(z_k|x_k) &= \mathcal{N}(z_k; H_kx_k, R_k)
\end{align*}
\]

- Non-linear non-Gaussian model: non-linear transformations & non-Gaussian noise

- e.g. Coordinated turn model & bearing only measurement

- e.g. constant velocity model & position observation
State-Space Models

Estimation: estimate (system) state/trajectory

- filtering: $\hat{x}_0, \ldots, \hat{x}_k$
- smoothing: $\hat{x}_{0:k}$

System Identification: estimate system parameters

Control: manipulate (system) state/trajectory
Bayes Filter

- **Hidden Markov Model**
  - States follow a 1\textsuperscript{st} order Markov process
    \[ p(x_k|x_{0:k-1}) = f_{k|k-1}(x_k|x_{k-1}) \], with initial pdf \( p_0(x_0) \)
    where \( x_{0:k} = [x_0, \ldots, x_k] \)
  - Observations are conditionally independent given \( x_{0:k} \)
    \[ p(z_{1:k}|x_{1:k}) = g_k(z_k|x_k) g_{k-1}(z_{k-1}|x_{k-1}) \cdots g_1(z_1|x_1) \]

- **Prior of trajectory**
  \[
p(x_0:k) = f_{k|k-1}(x_k|x_{k-1}) p(x_{0:k-1})
  = f_{k|k-1}(x_k|x_{k-1}) f_{k-1|k-2}(x_{k-1}|x_{k-2}) p(x_{0:k-2})
  = \prod_{i=1}^{k} f_{i|i-1}(x_i|x_{i-1}) p_0(x_0)
\]
Bayes Filter

- **Posterior PDF recursion** (smoothing-while-filtering)

\[ p(x_{0:k} \mid z_{1:k}) = p(x_{0:k-1} \mid z_{1:k-1}) \frac{g_k(z_k \mid x_k) f_{k|k-1}(x_k \mid x_{k-1})}{p(z_k \mid z_{1:k-1})}, \quad k \geq 1 \]

- **Filtering PDF recursion** (Bayes Filter)

\[ p_k(x_k \mid z_{1:k}) = \int p(x_{0:k} \mid z_{1:k})dx_{0:k-1} \]

**Prediction step**

\[ p_{k|k-1}(x_k \mid z_{1:k-1}) = \int f_{k|k-1}(x_k \mid x_{k-1}) p_{k-1}(x_{k-1} \mid z_{1:k-1})dx_{k-1} \]

**Update step**

\[ p_k(x_k \mid z_{1:k}) = \frac{g_k(z_k \mid x_k) p_{k|k-1}(x_k \mid z_{1:k-1})}{p(z_k \mid z_{1:k-1})} \]
Bayes filter

\[ \ldots \rightarrow p_{k-1}(\cdot \mid z_{1:k-1}) \quad \text{prediction} \quad \rightarrow p_{k|k-1}(\cdot \mid z_{1:k-1}) \quad \text{data-update} \quad \rightarrow p_{k}(\cdot \mid z_{1:k}) \quad \rightarrow \ldots \]

Kalman filter

\[ \ldots \rightarrow \mathcal{N}(\cdot; m_{k-1}, P_{k-1}) \quad \rightarrow \mathcal{N}(\cdot; m_{k|k-1}, P_{k|k-1}) \quad \rightarrow \mathcal{N}(\cdot; (m_k, P_k)) \quad \rightarrow \ldots \]

Particle filter

\[ \ldots \rightarrow \{ w_{k-1}^{(i)}, x_{k-1}^{(i)} \}_i^{N} \quad \rightarrow \{ w_{k|k-1}^{(i)}, x_{k|k-1}^{(i)} \}_i^{N} \quad \rightarrow \{ w_{k}^{(i)}, x_{k}^{(i)} \}_i^{N} \quad \rightarrow \ldots \]
Kalman Filter

Linear Gaussian model

\[
\begin{align*}
x_k &= F_{k|k-1} x_{k-1} + v_{k-1} \\
z &= H_k x_k + w_k
\end{align*}
\]

\[
\begin{align*}
f_{k|k-1}(x_k|x_{k-1}) &= \mathcal{N}(x_k; F_{k|k-1} x_{k-1}, Q_{k-1}) \\
g_k(z|x_k) &= \mathcal{N}(z; H_k x_k, R_k)
\end{align*}
\]

Kalman filter

\[
\begin{align*}
m_{k|k-1} &= F_{k|k-1} m_{k-1} \\
P_{k|k-1} &= F_{k|k-1} P_{k-1} F_{k|k-1}^T + Q_{k-1}
\end{align*}
\]

\[
\begin{align*}
m_k(z_k) &= m_{k|k-1} + K_k (z_k - H_k m_{k|k-1}) \\
P_k &= (I - K_k H_k) P_{k|k-1} \\
K_k &= P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}
\end{align*}
\]
Non-linear models

\[
x_k = \Phi_{k|k-1}(x_{k-1}) + v_{k-1} \\
z = \Psi_k(x_k) + w_k
\]

Extended Kalman filter (EKF):

Linearize model

\[
F_{k|k-1} = \nabla \Phi \bigg|_{x=\hat{x}_{k|k-1}} \\
H_k = \nabla \Psi \bigg|_{x=\hat{x}_{k|k-1}}
\]

Apply Kalman filtering equation to linearised model

\[
x_k \approx F_{k|k-1}x_{k-1} + v_{k-1} \\
z \approx H_kx_k + w_k
\]

Unscented Kalman filter (UKF)

Represent an \( n_x \)-dimensional Gaussian by \( 2n_x+1 \) sigma-points

Predicted PDF: approximately by applying \( \Phi_{k|k-1} \) to sigma-points & reconstruct Gaussian

Updated PDF: approximated by applying \( \Psi_k \) to sigma-points & reconstruct Gaussian.

Particle Filter

- Approximate posterior pdf by random samples (particles)
- Recursively generate particle approximation of posterior pdfs
- Virtually no assumption on the forms of

\[
\begin{align*}
  f_{k|k-1}(x_k|x_{k-1}) \\
  g_k(z_k|x_k)
\end{align*}
\]
Random Finite Set for Multi-object Dynamical System

- Introduction
- Review of Estimation Theory
  - Random variables
  - Estimator
  - State space models
  - Bayes filter
    - Kalman filter
    - Particle filter
- Multi-object Dynamical System
  - Multi-object (system) state
  - Multi-object (system) trajectory
  - Multi-object error metric
Multi-object State

Recall: Single-object (dynamical) system

\[ x_k = \Phi(x_{k-1}, u_{k-1}, n_k) \]

control
noise

\[ z_k = \Psi(x_k, u_{k-1}, v_k) \]

Markov transition density
Observation likelihood
Multi-object State

Multi-object System: Dynamical systems with multiple states

Fundamental difference from classical dynamical system:
- Random number of states and measurements
- False negatives, false positives, association uncertainty
What exactly is a **multi-target state**?

Can we stack individual states into a large **vector**?

**Error-metric**: **fundamental** in estimation/filtering & control

- What is the estimation error?

**Vector representation** does not admit multi-object error-metric

**Finite set representation** admits multi-object error-metric
Multi-object Trajectory

Recall: Fundamental (single-object) system problems

- **Estimation**: estimate (system) state/trajectory
  - filtering: $\hat{x}_0, \ldots, \hat{x}_k$
  - smoothing: $\hat{x}_{0:k}$

- **System Identification**: estimate system parameters

- **Control**: manipulate (system) state/trajectory
For single-object system: \textbf{trajectory} = \textit{history of states}

\[ x_{0:k} = [x_0, \ldots, x_k] \]

\begin{itemize}
  \item \textbf{multi-object (system) trajectory} = set of object trajectories
  \item \textbf{Multi-object trajectory} = history of multi-object states
\end{itemize}
Multi-object Trajectory

multi-object trajectory = history of labeled multi-object states

labeled multi-object state \( X = \{ x_1, \ldots, x_j, \ldots, x_n \} \)

labeled state vector \( (x_j, \ell_j) \in X \times L \)

kinematics label (distinct from other objects)

labeled trajectory = history of labeled state vectors with the same label
**Multi-object Trajectory**

**Labelling convention** (to ensure objects have distinct labels)

The state space is depicted over time, with objects born at time $k$ having distinct labels. Each object is labeled with $(k, i)$, where $k$ is the time of birth and $i$ is the object number. The space of labels for objects born at time $k$ is given by $L_k = \{k\} \times \mathbb{N}$. The space of all labels at time $k$ can be expressed as $L = L_{0:k} = L_{0:k-1} \cup L_k$. The labeled trajectories are shown as continuous lines connecting the state space over time.
Fundamental multi-object system problems

- **Estimation**: estimate multi-object state/trajectory
  - filtering with **labeled** multi-object states: \( \hat{X}_0, \ldots, \hat{X}_k \)
  - smoothing with **labeled** multi-object states: \( \hat{X}_{0:k} \)

- **System Identification**: estimate multi-object system parameters

- **Control**: manipulate multi-object state/trajectory
Naïve formulation: finite subset of the disjoint union $\bigcup_{n} N \times X^n$

[García-Fernández et. al. “Trajectory probability hypothesis density filter” FUSION 2018]

Nonsensical scenario: this formulation allows the set of trajectories

\[
\left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix} \right\}
\]

which consist of 3 distinct elements from the space of “trajectories”, … but they correspond to different segments of the same trajectory.

Unable to

- estimate the set of trajectories by filtering (of the multi-object state)
- provide ancestry or lineage information for spawning objects
- truncate posterior density (essential for direct approximations) … no longer a function of sets

In practice a tracking system is required to assign labels to tracks/trajectories
Multi-object Error Metric

- Properties of an error metric that are taken for granted
- **Metric**: $d(\cdot,\cdot)$
  - (identity) $d(x, y) = 0$ iff $x = y$;
  - (symmetry) $d(x, y) = d(y, x)$ for all $x, y$
  - (triangle inequality) $d(x, y) \leq d(x, z) + d(z; y)$ for all $x, y, z$.

Why triangle inequality?

Suppose estimate $z$ is “close” to the true state $x$.
If estimate $y$ is “close” to $z$, then $y$ is also “close” to $x$

- Multi-object metric must also
  - capture errors in number of objects and state (position) errors
  - have meaningful interpretation
Meaningful interpretation: when cardinality estimate is correct

(a) 1 object with 1m error

(b) 100 objects with 0.1m error each

Which estimate is more accurate, (a) or (b)?

**Much harder** to estimate more objects with better accuracy

Meaningful multi-object metric must declare (b) more accurate than (a)
Multi-object Error Metric

Meaningful interpretation: when cardinality estimate is not necessarily correct

Which estimate is more accurate?

- (a)
- (b)
- (c)

○: True positions
+: Estimated positions

Not obvious
Multi-object Error Metric

Optimal Sub-Pattern Assignment (OSPA) distance between 2 finite sets $X$ and $Y$

Base distance (distance between 2 points): $d(x, y)$

$$d^{(c)}(x, y) = \min\{d(x, y), c\}$$

$$d^{(c,p)}([x_1, ..., x_n], [y_1, ..., y_n]) = (d^{(c)}(x_1, y_1)^p + ... + d^{(c)}(x_n, y_n)^p)^{1/p}$$

$$d_{OSPA}^{(c,p)}(X, Y) = \left( \min_{ perm(Y) } d^{(c,p)}(X', perm(Y))^p \right)^{1/p}$$

Interpretation: “best case cardinality + state error, per object”

Efficiently computed by the Hungarian algorithm
Multi-object Error Metric

Optimal Sub-Pattern Assignment (OSPA) distance between 2 finite sets $X$ and $Y$

$$X = \{x_1, ..., x_m\}, Y = \{y_1, ..., y_n\}$$

$$d^{(c,p)}_{OSPA}(X,Y) = \begin{cases} 0, & X = Y \\ \left( \frac{1}{n} \min_{\pi} \sum_{i=1}^{m} d^{(c)}(x_{i}, y_{\pi(i)})^p + c^p (n-m) \right)^{1/p}, & m \leq n \\ d^{(c,p)}_{OSPA}(Y,X), & m > n \end{cases}$$

$$d^{(c)}(x, y) = \min(d(x, y), c)$$

$\pi = permutation$


- Per object normalization, i.e. normalization by $n$ is crucial
- Variations of OSPA without per object normalization:
  - COLA (for SLAM): divide unnormalized result by $c$ [Barios et. al. 15, 17]
  - GOSPA (for tracking): scaling cardinality [Rahmathullah et. al. 17]
Test for meaningful multi-object metric

(a) 1 object with 1m error

(b) 100 objects with 0.1m error each

Hausdorff = Wasserstein = OSPA = 1m (using p=1)
GOSPA = 1m

Hausdorff = Wasserstein = OSPA = 0.1m
GOSPA = 10m

Hausdorff, Wasserstein, OSPA all suggest (b) is 10 times more accurate than (a)
GOSPA suggests (b) is 10 times less accurate than (a) …nonsensical
Multi-object Error Metric

- Another test scenario for meaningfulness of multi-object metric
  No. targets at time $k$: $10^{2k}$, $k = 0,1,\ldots$ exponential growth
  A (very good) multi-target filter achieves the following results:
    Estimated no. targets at time $k$: $10^{2k}$, $k = 0,1,\ldots$ (perfect cardinality estimates)
    Error for each target time $k$ (when optimally matched): $10^{-k}$, $k = 0,1,\ldots$

- Common sense: the multi-object filter is doing better and better with time:
  Correctly estimate larger and larger no. targets with better and better accuracy.
  Perfect estimate as $k$ tends to infinity

- What are the metrics saying?
  OSPA error $= 10^{-k}$ … approaches 0 as $k$ tends to infinity … agrees with intuition
  Unnormalized OSPA error $= 10^{k}$ … explodes as $k$ tends to infinity! …

  No per object normalization … nonsensical multi-object estimation error!
What about multi-object trajectory estimation error?

State space trajectories multi-object states

multi-object (system) trajectory = set of object trajectories

OSPA distance between 2 finite sets of tracks

Need a suitable base distance between 2 tracks
A **track** is a mapping $f_T: \{1,2,\ldots,K\} \rightarrow$ (single-object) state space

- The domain $T$ of $f_T$ is the set of times at which the object exists
- Permits fragmented tracks, i.e. track with holes
- Many tracking algorithms do not always produce continuous tracks

A **trajectory** is a non-fragmented track, i.e. track without holes

Multi-object Error Metric

Suitable base distance between 2 tracks

\[ d^{(c)}(g_S, f_T) = \frac{1}{|S \cup T|} \sum_{t \in S \cup T} d^{(c,p)}_{OSPA}(\{g_S(t)\}, \{f_T(t)\}) \]

Extension of the traditional MSE to tracks of different supports
Multi-object Error Metric

\[ \text{OSPA}^{(2)} \quad d^{(c,p)} = \text{OSPA with base distance } d^{(c)} \]

- Distance between 2 sets of tracks (satisfies metric properties)
- OSPA on OSPA: OSPA distance with OSPA base distance
- Best case MSE per track

Displaying OSPA\(^{(2)}\) tracking error with a moving window

- Same as OSPA when \(L=1\)
- Forgets states outside window
- Possible windowing artifact

Scalable (millions of tracks)
Random Finite Set for Multi-object Dynamical System

- **Bayesian Multi-object Estimation**
  - Random Finite Set
  - Probability Density of an RFS
  - Multi-object Estimators
  - Multi-object State Space Model
  - Multi-object Filtering

- **Generalized Labeled Multi-Bernoulli (GLMB) Filter**
  - Labeled RFS
  - GLMB
  - GLMB Filter
    - GLMB Prediction Truncation
    - GLMB Update Truncation
    - Joint GLMB Prediction and Update
Random Finite Set

- Multi-object state: Finite set
- Multi-object System: set-valued dynamical system

Stochastic system models state/observation as random variables

- Requires the concept of random finite set (RFS)
- Requires the concept of integration/density for set-valued variable

What’s the big deal about working with sets?

Most widely adopted practice (in engineering/computer science):

\[ X = \{x_1, \ldots, x_m\} \quad \rightarrow \quad p(X) = p(x_1, \ldots, x_m) \]

- **Object Tracking / Computer Vision**: \( X = \) set of object states or tracks
- **Simultaneous Localization & Mapping**: \( X = \) set of landmarks
- **Multiple Instance (MI) Learning**: \( X = \) set of features/instances

\[ p(X \mid \text{Data}) \quad \text{Posterior PDF} \]
\[ p(\text{Data} \mid X) \quad \text{Data likelihood function} \]
**Random Finite Set**

What’s the big deal about working with sets?

- Apples land on the ground independently from each other

\[ p(X) = p(x_1, ..., x_m) = \prod_{i=1}^{m} p_f(x_i) \]

- Daily landing patterns are independent from each other

**Novelty Detection:** find unlikely daily landing patterns
What’s the big deal about working with sets?

$p_f$ (PDF of landing positions learned from “normal” training data)

Day 1: $x_1$

$$p(x_1) = p_f(x_1) = 0.2,$$

Day 2: $x_2, x_3$

$$p(x_2, x_3) = p_f(x_2) p_f(x_3) = 0.36$$

- Q: Which pattern is less likely? Ans: day 1 pattern is less likely than day 2.
- Change unit of measurement from m to cm and ...

$$p(x_1) = 0.002 > p(x_2, x_3) = 0.000036$$
Random Finite Set

What’s the big deal about working with sets?

- Can’t treat a finite set as if it were a vector!
- Naïve application of random vector PDF suffers from:
  - Inconsistency with units of measurement
  - No suitable cardinality information
- Multi-object state space model (HMM) requires:
  - Markov transition density for finite-set-valued state
  - Observation likelihood for finite-set-valued observation
  - Need PDFs for Random finite sets

Inconsistent PDFs inference?
What is a random finite set (RFS)?

- The number of points is random,
- The points have no ordering and are random
- An RFS is a finite set-valued random variable
- Also known as: (simple finite) point process
Random Finite Set

**Bernoulli RFS**

\[
sample \ u \sim \text{uniform}[0,1] \\
\text{if} \ u < r, \\
sample \ x \sim p(\cdot), \\
\text{end;}
\]

Completely characterised by the parameter pair \((r, p)\)

**Binomial RFS**

\[
\text{for } i=1:n, \\
sample \ x_i \sim p(\cdot), \\
\text{end;}
\]

Completely characterised by the parameter pair \((n, p)\)
Random Finite Set

Poisson RFS

Sample $n \sim \text{Pois}(r)$,
for $i=1:n$,
    sample $x_i \sim p(\cdot)$,
end;

Completely characterised by the product $rp$

i.i.d. cluster RFS

Sample $n \sim c(\cdot)$,
for $i=1:n$,
    sample $x_i \sim p(\cdot)$,
end;

Completely characterised by the parameter pair $(c, p)$
Probability Density of an RFS

Probability distribution of $\Sigma$

$P_\Sigma(T) = P(\Sigma \in T), \ T \subseteq F(X)$

Belief “distribution” of $\Sigma$

$\beta_\Sigma(T) = P(\Sigma \subseteq T), \ T \subseteq X$

Collection of finite subsets of $X$
Probability Density of an RFS

**Point Process Theory (1950-1960’s)**

**Probability density** of $\Sigma$

$$p_\Sigma : \mathcal{F}(X) \rightarrow [0, \infty)$$

$$P_\Sigma (T) = \int_T p_\Sigma (X) \mu(dX)$$

**Belief “distribution”** of $\Sigma$

$$\beta_\Sigma (T) = P(\Sigma \subseteq T) , T \subseteq X$$

**Probability distribution** of $\Sigma$

$$P_\Sigma (T) = P(\Sigma \in T) , T \subseteq \mathcal{F}(X)$$

Collection of finite subsets of $X$

State space

X

T

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$

$\Sigma$
Recap: Probability distribution & density on Euclidean space

For $\mathcal{T} \subseteq F(X)$ what is its volume? Need to resort to the more general notion of density!

Probability per unit volume:

$$p_X(x) = \lim_{\text{Vol}(\Delta_x) \to 0} \frac{P_X(\Delta_x)}{\text{Vol}(\Delta_x)} = \frac{P_X(dx)}{dx}$$

$$P_X(S) = \int_S p_X(x)dx$$

Density w.r.t. a measure $\mu$

$$p_X(x) = \lim_{\mu(\Delta_x) \to 0} \frac{P_X(\Delta_x)}{\mu(\Delta_x)} = \frac{P_X(dx)}{\mu(dx)}$$

$$P_X(S) = \int_S p_X(x)\mu(dx)$$
Integration on $F(\mathbf{X})$: Given a measure $\mu$ on $F(\mathbf{X})$, the integral of a staircase function $f : F(\mathbf{X}) \rightarrow [0, \infty)$ defined by $f(X) = \sum_i f_i 1_{T_i}(X)$ is

$$\int f(X) \mu(dX) = \sum_i f_i \mu(T_i)$$

some measure of the size of $T_i$

For a more general function $f$:

use the staircase functions to approximate $f$

integral of $f = \text{limit of integrals of the staircase functions}$
**Probability Density of an RFS**

- Integration on $\mathcal{F}(\mathbb{X})$: Given a measure $\mu$ on $\mathcal{F}(\mathbb{X})$:

  $$\int_{\mathcal{T}} f(X) \mu(dX) = \int_{\mathcal{T}} 1_{\mathcal{T}}(X) f(X) \mu(dX)$$

- The PDF of an RFS $\Sigma$ relative to $\mu$ is a function $p_\Sigma : \mathcal{F}(\mathbb{X}) \rightarrow [0, \infty)$ satisfying:

  $$P_\Sigma(T) = \int_T p_\Sigma(X) \mu(dX),$$

  Unless $\mathbb{X} = \mathbb{R}^n$ (discrete space), there is no proof that such PDF exists.

- Standard reference measure for RFS:

  $$\mu(T) = \sum_{k=0}^{\infty} \frac{1}{k! K^k} \int_X \cdots \int_X 1_T(\{x_1, \ldots, x_k\}) dx_1 \cdots dx_k$$

  unit in which volume on $\mathbb{X}$ is measured in

  $$\int_{\mathcal{T}} f(X) \mu(dX) = \sum_{k=0}^{\infty} \frac{1}{K^k k!} \int_X \cdots \int_X 1_T(\{x_1, \ldots, x_k\}) f(\{x_1, \ldots, x_k\}) dx_1, \ldots, dx_k$$
Probability Density of an RFS

Point Process Theory (1950-1960’s)

Belief “distribution” of $\Sigma$
$$\beta_{\Sigma}(T) = P(\Sigma \subseteq T), \ T \subseteq \mathbb{X}$$

Mahler’s Finite Set Statistics (1994)

Belief “density” of $\Sigma$
$$\beta_{\Sigma}(T) = \int_{T} f_{\Sigma}(X) \delta X$$

Probability distribution of $\Sigma$
$$P_{\Sigma}(T) = P(\Sigma \in T), \ T \subseteq \mathcal{F}(\mathbb{X})$$

Conventional integral

Probability density of $\Sigma$
$$p_{\Sigma}(X) : \mathcal{F}(\mathbb{X}) \rightarrow [0, \infty)$$
$$P_{\Sigma}(T) = \int_{T} p_{\Sigma}(X) \mu(dX)$$

Set integral

Collection of finite subsets of $\mathbb{X}$

State space

$\sum$

$\mathcal{T}$

$\mathcal{X}$

$\mathcal{T}$
Probability Density of an RFS

Set integral of $f : \mathcal{F}(\mathbb{X}) \to [0, \infty)$, on $S$ in $\mathcal{C}(\mathbb{X})$ (the closed subsets of $\mathbb{X}$)

$$
\int_S f(X) \delta X = \int_{X \subseteq S} f(X) \delta X \equiv \sum_{k=0}^{\infty} \frac{1}{k!} \int_S \ldots \int_S f(\{x_1, \ldots, x_k\}) \, dx_1 \ldots dx_k
$$

Set derivative of $F : \mathcal{C}(\mathbb{X}) \to [0, \infty)$, at a point in $\mathcal{F}(\mathbb{X})$

$$(\delta F)_\emptyset (S) = F(S)$$

$$(\delta F)_{\{x\}} (S) = \lim_{\text{vol}(\Delta_x) \to 0} \frac{F(S \cup \Delta_x) - F(S)}{\text{vol}(\Delta_x)}$$

$$(\delta F)_{\{x_1, \ldots, x_m\}} (S) = \delta((\delta F)_{\{x_1, \ldots, x_{m-1}\}})_{\{x_m\}} (S)$$

$\Delta_x = \text{neighbourhood of } x$

Fundamental Theorem of FISST Calculus

$$f(X) = (\delta F)_X(\emptyset) \iff F(S) = \int_S f(X) \delta X$$

Probability Density of an RFS

\[
P_\Sigma(T) = \int_{\tau} p_\Sigma(X) \mu(dX) \sum_{k=0}^{\infty} \frac{1}{K^k k!} \int_{X} \cdots \int_{X} 1_{\tau}(\{x_1, \ldots, x_k\}) p_\Sigma(\{x_1, \ldots, x_k\}) dx_1, \ldots, dx_k
\]

\[
\beta_\Sigma(S) = \int_{X \subseteq S} f_\Sigma(X) \delta X \sum_{k=0}^{\infty} \frac{1}{k!} \int_{S} \cdots \int_{S} f_\Sigma(\{x_1, \ldots, x_k\}) dx_1 \ldots dx_k
\]

\[
p_\Sigma(X) = K^{\|X\|} f_\Sigma(X)
\]

\[
\int f(X) \mu(dX) = \int K^{-\|X\|} f(X) \delta X
\]

⇒ conditional densities, total probability, Bayes rule hold for belief “densities”

Probability Density of an RFS

**Point Process Theory (1950-1960’s)**

**Probability distribution** of $\Sigma$

$$P_\Sigma (\mathcal{T}) = P(\Sigma \in \mathcal{T}), \mathcal{T} \subseteq \mathcal{F}(\mathbb{X})$$

**Probability density** of $\Sigma$

$$p_\Sigma : \mathcal{F}(\mathbb{X}) \to [0, \infty)$$

$$P_\Sigma (\mathcal{T}) = \int_\mathcal{T} p_\Sigma (X) \mu(dX)$$

**Choquet (1968)**

**Belief “distribution”** of $\Sigma$

$$\beta_\Sigma (\mathcal{T}) = P(\Sigma \subseteq \mathcal{T}), \mathcal{T} \subseteq \mathbb{X}$$

**Belief “density”** of $\Sigma$

$$f_\Sigma : \mathcal{F}(\mathbb{X}) \to [0, \infty)$$

$$\beta_\Sigma (\mathcal{T}) = \int_\mathcal{T} f_\Sigma (X) \delta X$$

**Mahler’s Finite Set Statistics (1994)**

**Conventional integral**

**Collection of finite subsets of $\mathbb{X}$**

**State space**

**VSD (2005)**

**Set integral**
Multi-object Estimator

Data $Z_{1:k} = [Z_1, ..., Z_k]$ (can be an array of finite sets)

Data model: $p(Z_{1:k} \mid \Theta) =$ Probability density of data $Z_{1:k}$, given finite-set-valued parameter $\Theta$

Likelihood function

Estimator

$\hat{\Theta} : Z \mapsto \hat{\Theta}(Z_{1:k}) \in \text{space of finite sets}$

estimate
**Bayes Risk:** Expected posterior cost/penalty of incorrect estimate

\[
R(\tilde{\Theta}) = E \left[ C(\tilde{\Theta}(Z_{1:k}), \Theta) \right] = \int C(\tilde{\Theta}(Z_{1:k}), \Theta) p(Z_{1:k} | \Theta) p(\Theta) \, d\Theta \, dZ_{1:k}
\]

Bayes risk  
Penalty of using \(\tilde{\Theta}(Z_{1:k})\) as an estimate of \(\Theta\)  
set integrals

**Optimal Bayes estimator:**

\[
\hat{\Theta}^{Bayes} : Z_{1:k} \mapsto \hat{\Theta}^{Bayes} (Z_{1:k}) = \arg \min_{\Theta} R(\tilde{\Theta} | Z_{1:k})
\]

Posterior mean \(E[\Theta | Z_{1:k}]\) ???
Multi-object Estimator

Joint multi-object estimator: given a dimensionless constant $D$

$$\hat{\Theta}_{D}^{JoM} = \arg \sup_{\Theta} p(\Theta \mid Z_{1:k}) \frac{D^{\mid \Theta \mid}}{\mid \Theta \mid !}$$

- **Converges** as $k$ tends to infinity
- The constant $D$ determines the rate of convergence

Marginal Multi-object estimator:

$$\hat{n} = \arg \sup_{n} p(|\Theta| = n \mid Z_{1:k})$$

$$\hat{\Theta}_{\hat{n}}^{MaM} = \arg \sup_{\Theta: |\Theta| = \hat{n}} p(\Theta \mid Z_{1:k})$$

Multi-object State Space Model

Multi-object Dynamic Model

\[ X_k = S_{k|k-1}(X_{k-1}) \cup B_{k|k-1}(X_{k-1}) \cup \Gamma_k \]

Multi-object density transition

\[ f_{k|k-1}(X_k | X_{k-1}) \]

Survival Probability

\[ P_{S,k}(x_{k-1}) \]

Single-object Transition

\[ f_{k|k-1}(x_k | x_{k-1}) \]

Spawn RFS

\[ B_{k|k-1}(X_{k-1}) \]

Birth RFS

\[ \Gamma_k \]

Evolution of each element \( x \) of a given multi-object state \( X_{k-1} \)
Multi-object State Space Model

Multi-object Dynamic Model

\[ X_k = S_{k|k-1}(X_{k-1}) \cup B_{k|k-1}(X_{k-1}) \cup \Gamma_k \]

\[ f_{k|k-1}(X_k \mid X_{k-1}) = \left( p_{S_{k|k-1}} \left( \cdot \mid X_{k-1} \right) \ast p_{B_k} \left( \cdot \mid X_{k-1} \right) \ast p_{\Gamma_k} \right)(X_k) \]
Multi-object State Space Model

Multi-object Measurement Model

\[ Z_k = \Theta_k(X_k) \cup K_k \]

Multi-object likelihood

\[ g_k(Z_k|X_k) \]

Detection Probability

\[ P_{D,k}(x) \]

Single-object Likelihood

\[ g_k(z|x) \]

Clutter Intensity

\[ \kappa_k(z) \]

Observation process for each element \( x \) of a given multi-object state \( X_k \)
Multi-object State Space Model

Multi-object Measurement Model

\[ Z_k = \Theta_k(X_k) \cup K_k \]

\[ g_k(Z_k | X_k) = \sum_{W \subseteq Z_k} \left( p_{\Theta_k} (W | X_k) p_{K_k} (Z_k - W) \right) \]

\[ = \left( p_{\Theta_k} (\cdot | X_k) * p_{K_k} (\cdot) \right) (Z_k) \]
Multi-object State Space Model

Multi-object likelihood

\[ \Psi^{(j)}_{\{z_1, \ldots, z_m\}, k}(x) = \begin{cases} \frac{P_{D,k}(x) g_k(z_j | x)}{\kappa_k(z_j)}, & j > 0 \\ 1 - P_{D,k}(x), & j = 0 \end{cases} \]

\[ g_k(\{z_1, \ldots, z_m\} \mid \{x_1, \ldots, x_n\}) \propto \sum_{\theta \in \Theta_k(n,m)} \prod_{i=1}^{n} \Psi^{(\theta(i))}_{\{z_1, \ldots, z_m\}, k}(x_i) \]

space of association maps \( \theta : \{1, \ldots, n\} \rightarrow \{0, \ldots, m\} \)

such that \( \theta(i) = \theta(i') > 0 \Rightarrow i = i' \)

association map assigns at most 1 measurement index to an object
Multi-object Filtering

Multi-target Bayes filter

\[ \cdots \rightarrow p_{k-1}(X_{k-1}\mid Z_{1:k-1}) \quad \xrightarrow{\text{prediction}} \quad p_{k|k-1}(X_k\mid Z_{1:k}) \quad \xrightarrow{\text{data-update}} \quad p_k(X_k\mid Z_{1:k}) \quad \rightarrow \cdots \]

\[ \int f_{k|k-1}(X_k \mid X)p_{k-1}(X \mid Z_{1:k-1})\delta X \]

\[ \frac{g_k(Z_k \mid X_k)p_{k|k-1}(X_k \mid Z_{1:k-1})}{\int g_k(Z_k \mid X)p_{k|k-1}(X \mid Z_{1:k-1})\delta X} \]

- Computationally expensive in general!

Single-target

- State of system: random vector
- Single-target Bayes filter
- First-moment filter (e.g. \(\alpha-\beta-\gamma\) filter)

Multi-target

- State of system: random set
- Multi-target Bayes filter
- First-moment filter (“PHD” filter)

Multi-object Filtering

PHD (intensity function)

\[ \nu_\Sigma(x) = \mathbb{E}[\delta_\Sigma(x)] \]

\[ \nu_\Sigma(x) = \int p_\Sigma(\{x\} \cup Y) \delta Y \]
Multi-object Filtering

PHD (intensity function)

\[ \int_S v_\Sigma(x) dx = \text{expected number of points in } S \]

\[ v_\Sigma(x) = \text{density of expected number of points at } x \]

or expected number of points per unit volume at \( x \).

Local maxima of intensity gives estimates of the points.
**Multi-object Filtering**

\[
\cdots \rightarrow p_{k-1}(X_{k-1}|Z_{1:k-1}) \quad \text{prediction} \quad \rightarrow p_{k|k-1}(X_k|Z_{1:k-1}) \quad \text{update} \quad \rightarrow p_k(X_k|Z_{1:k}) \quad \rightarrow \cdots
\]

Multi-object Bayes filter

\[
\cdots \rightarrow \nu_{k-1}(x_{k-1}|Z_{1:k-1}) \quad \text{PHD prediction} \quad \rightarrow \nu_{k|k-1}(x_k|Z_{1:k-1}) \quad \text{PHD update} \quad \rightarrow \nu_k(x_k|Z_{1:k}) \quad \rightarrow \cdots
\]

PHD filter: 1\textsuperscript{st} moment (Poisson) approximation


**PHD filter implementations:**


**Multi-object Filtering**

- **Drawback of PHD filter**: High variance of cardinality estimate
- **Relax Poisson assumption**: allows any cardinality distribution
- **Jointly propagate**: PHD & cardinality distribution.

---


**Multi-object Filtering**

**Multi-Bernoulli Filter**

- **Propagates:** parameterized approximation to posterior density

\[
\cdots \rightarrow \left\{ (r_{k-1}^{(i)}, p_{k-1}^{(i)}) \right\}_{i=1}^{M_{k-1}} \xrightarrow{\text{prediction}} \left\{ (r_{k|k-1}^{(i)}, p_{k|k-1}^{(i)}) \right\}_{i=1}^{M_{k|k-1}} \xrightarrow{\text{update}} \left\{ (r_k^{(i)}, p_k^{(i)}) \right\}_{i=1}^{M_k} \rightarrow \cdots
\]

- **MB:** Multi-Bernoulli filter
- **hypothesized tracks:** existence prob + spatial density


Multi-object Filtering

Multi-Bernoulli filter for image data

- Estimate number of targets & their states directly from image data
- Track-Before-Detect
- Computer vision

Multi-object Bayes filter for image data

\[ \cdots \xrightarrow{\text{prediction}} p_{k-1}(X_{k-1}|y_{1:k-1}) \xrightarrow{\text{update}} p_{k|k-1}(X_k|y_{1:k}) \xrightarrow{\text{prediction}} \cdots \]

\[ \cdots \xrightarrow{\text{prediction}} \{(r_{k-1}^{(i)}, p_{k-1}^{(i)})\}_{i=1}^{M_{k-1}} \xrightarrow{\text{update}} \{(r_k^{(i)}, p_{k|k-1}^{(i)})\}_{i=1}^{M_{k|k-1}} \xrightarrow{\text{prediction}} \{(r_k^{(i)}, p_k^{(i)})\}_{i=1}^{M_k} \xrightarrow{\text{update}} \cdots \]

Multi-object Filtering

**Caution:** Truncating a multi-object density may not give a multi-object density.

Multi-object likelihood:

\[
g(\{z_i\}|\{x_1, x_2\}) \propto P_D(x_1)g(z_i|x_1)(1-P_D(x_2)) + (1-P_D(x_1))P_D(x_2)g(z_i|x_2) + (1-P_D(x_1))(1-P_D(x_2))\kappa(z_i)
\]

- \(x_1\) detected, \(x_2\) missed
- \(x_1\) missed, \(x_2\) detected
- \(x_1\) & \(x_2\) missed

Multi-object posterior for the prior \(p(\{x_1, x_2\})\):

\[
p(\{x_1, x_2\}|\{z_i\}) \propto P_D(x_1)g(z_i|x_1)(1-P_D(x_2))p(\{x_1, x_2\})
\]

\[
+ (1-P_D(x_1))P_D(x_2)g(z_i|x_2)p(\{x_1, x_2\}) + (1-P_D(x_1))(1-P_D(x_2))\kappa(z_i)p(\{x_1, x_2\})
\]

Suppose we truncate the last 2 terms

\[
f(x_1, x_2|\{z_i\}) \square P_D(x_1)g(z_i|x_1)(1-P_D(x_2))p(\{x_1, x_2\})
\]

Truncated function \(f\) is not symmetric in its arguments

\[
f(a,b|\{z_i\}) = P_D(a)g(z_i|a)(1-P_D(b))p(\{a,b\})
\]

\[
f(b,a|\{z_i\}) = P_D(b)g(z_i|b)(1-P_D(a))p(\{a,b\})
\]

\[
f(a,b|\{z_i\}) \neq f(b,a|\{z_i\})
\]

\(f\) is not a function of the set \(\{a,b\}\), hence cannot be a multi-object density